# Unüberwachtes Lernen mit dem Herz-Datensatz

## Clustering und PCA auf die Herzdaten

### Einlesen der Herz-Daten

Es werden wieder die Herzdaten aus der letzten Aufgabe genutzt. Lesen Sie diese als Data Frame ein.

```
df = read.csv(url('https://oc.informatik.hs-mannheim.de/s/wyzFq34K9HiNjXR/download'))
```

### Bedeutet "ähnliche Merkmale" auch "gleiche Diagnose"?

Für jeden Datensatz ist bekannt, zu welcher Klasse er gehört: 0 (gesund) und 1 (erkrankt). Wir wollen untersuchen, wie gut *ähnliche* Datensätze zur gleichen Klasse gehören. Dafür soll mit dem $k$-means-Clusterverfahren der Datensatz in zwei Cluster eingeteilt werden.

#### Nur reelle Merkmale

Zunächst sollen **nur die nummerischen Merkmale** benutzt werden und nicht jene, die Faktoren sind.

**Clustering** Clustern Sie diese Daten. Überlegen Sie, ob Sie die Daten standardisieren wollen.

```
print(head(df[sapply(df, is.numeric)]))
```

```
##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope goal
## 1  63   1  1      145  233   1       2     150     0     2.3     3    0
## 2  67   1  4      160  286   0       2     108     1     1.5     2    2
## 3  67   1  4      120  229   0       2     129     1     2.6     2    1
## 4  37   1  3      130  250   0       0     187     0     3.5     3    0
## 5  41   0  2      130  204   0       2     172     0     1.4     1    0
## 6  56   1  2      120  236   0       0     178     0     0.8     1    0
```

```
# numerische Merkmale: age, trestbps, chol, thalach, oldpeak, ca
# faktorielle Merkmale: sex, cp, fbs, restecg, exang, slope, thal, goal
# Merkmale initial nicht richtig zugeordnet -> umwandeln
conv_to_factor = c("sex", "cp", "fbs", "restecg", "exang", "slope", "goal")
df_clust = df %>% mutate_at(conv_to_factor, factor)
df_clust$ca = as.numeric(as.factor(df_clust$ca))

# add df for healthy/sick (1= healtyh, 0=sick)
healthy = df_clust$goal!=0

# select numerical values
df_nums = df_clust %>% select_if(is.numeric)

# scale data (features have different scales
# -> features have different influence on distance calculation)
df_nums = data.frame(scale(df_nums))

# clustering, nstart is the number of random initial cluster centers
```

```r
set.seed(42)
km_res = kmeans(df_nums, centers=2, nstart=10)
```

**Richtig?** Berechnen Sie, wie viel Prozent der Datensätze richtig einem Cluster eingeordnet wurden und geben Sie die Zahl auf zwei Nachkommastellen gerundet aus.

Hinweis: Berücksichtigen Sie, dass die Vergabe der Clusternummern zufällig ist. D.h. sowohl die Cluster (1, 2) wie auch (2, 1) sind möglich.

```r
cluster1 = sapply(km_res$cluster, function(X) X==1)
cluster2 = sapply(km_res$cluster, function(X) X==2)

perc_cluster1 = round(mean(cluster1 == healthy) * 100, digits = 2)
perc_cluster2 = round(mean(cluster2 == healthy) * 100, digits = 2)

# clusters have no semantic meaning (healthy/sick) -> choose cluster with higher correspondence
if (perc_cluster1 > perc_cluster2) {
    km_healthy = cluster1
    max_perc = perc_cluster1
} else {
    km_healthy = cluster2
    max_perc = perc_cluster2
}

print(paste(max_perc, "% der Datensätze wurden richtig einem Cluster zugeordnet"))
```

```
## [1] "74.59 % der Datensätze wurden richtig einem Cluster zugeordnet"
```

**Scatterplot age vs. thalach** Plotten Sie die Merkmale `age` und `thalach` als Scatterplot. Färben Sie die Punkte gemäß ihrer Clusterzuordnung ein. Die Form (`shape`) eines Punkts soll zeigen, ob die Klassifikation (d.h. der Cluster) richtig oder falsch ist.

```r
df_nums$cluster = as.factor(km_res$cluster)
df_nums$correct = km_healthy == healthy

ggplot(df_nums) +
    geom_point(aes(x=age, y=thalach, color=cluster, shape=correct), size=3)
```


**Mit Dummy-Variablen**

Nun sollen **alle Merkmale** benutzt werden.

**Clustering** Clustern Sie diese Daten. Überlegen Sie, wie die Faktoren zu Zahlen werden.

```
# convert factors to numeric
df_dummy = df %>% mutate_if(function(x) !is.numeric(x), function(y) as.numeric(as.factor(y)))

# scale data
df_dummy = data.frame(scale(df_dummy))

# clustering
km_res = kmeans(df_dummy, centers=2, nstart=10)
```

**Richtig?** Berechnen Sie für diesen Fall, wie viel Prozent der Datensätze richtig einem Cluster eingeordnet wurden und geben Sie die Zahl auf zwei Nachkommastellen gerundet aus. Wie hat sich der Wert verändert? Warum ist dies so?

```
cluster1 = sapply(km_res$cluster, function(X) X==1)
cluster2 = sapply(km_res$cluster, function(X) X==2)

perc_cluster1 = round(mean(cluster1 == healthy) * 100, digits = 2)
perc_cluster2 = round(mean(cluster2 == healthy) * 100, digits = 2)

if (perc_cluster1 > perc_cluster2) {
    km_healthy = cluster1
```

```
    max_perc = perc_cluster1
} else {
    km_healthy = cluster2
    max_perc = perc_cluster2
}

print(paste(max_perc, "% der Datensätze wurden richtig einem Cluster zugeordnet"))
```

```
## [1] "85.15 % der Datensätze wurden richtig einem Cluster zugeordnet"
```

**Scatterplot age vs. thalach** Plotten Sie erneut und schauen Sie, wie die richtigen nun Punkte verteilt sind.

```
df_dummy$cluster = as.factor(km_res$cluster)
df_dummy$correct = km_healthy == healthy

ggplot(df_dummy) +
    geom_point(aes(x=age, y=thalach, color=cluster, shape=correct), size=3) +
    scale_color_discrete(drop = FALSE)
```


## PCA

Wenden Sie eine PCA auf diesen Datensatz an. Es sollen alle Merkmale berücksichtigt werden.

### Wichtige Merkmale

Welche Merkmale der ersten Hauptkomponente tragen am meisten zur Varianz bei? Geben Sie die TOP-10-Merkmale an.

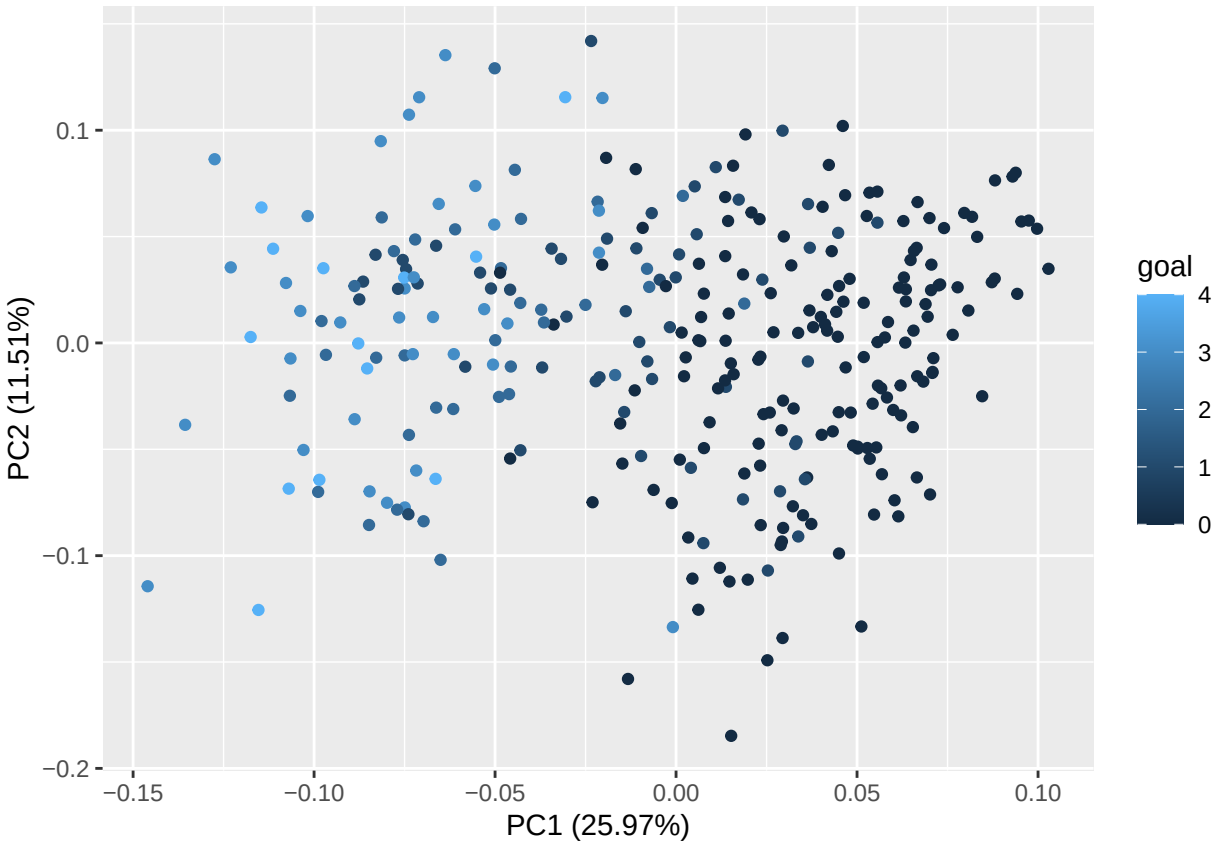
```r
# pca only works with numerical values -> all features should be considered
# -> convert to numeric values
df_pca = df %>% mutate_if(function(x) !is.numeric(x), function(y) as.numeric(as.factor(y)))

# apply pca
pca_res = prcomp(df_pca, scale=TRUE)

# positive values: positive correlation
# negative values: negative correlation
pc1 = pca_res$rotation
first_comp = data.frame(pca_res$rotation[, 1])

# the higher the absolute value, the more it contributes to the variance
first_comp_desc = abs(first_comp) %>% arrange(desc(pca_res.rotation...1.))
head(first_comp_desc,10)
```

```
##          pca_res.rotation...1.
## goal                 0.4253600
## oldpeak              0.3633676
## thalach              0.3426867
## thal                 0.3172153
## slope                0.3113089
## exang                0.3021607
## ca                   0.2994957
## cp                   0.2686758
## age                  0.2397569
## trestbps             0.1398275
```

### Erste und zweite Hauptkomponente

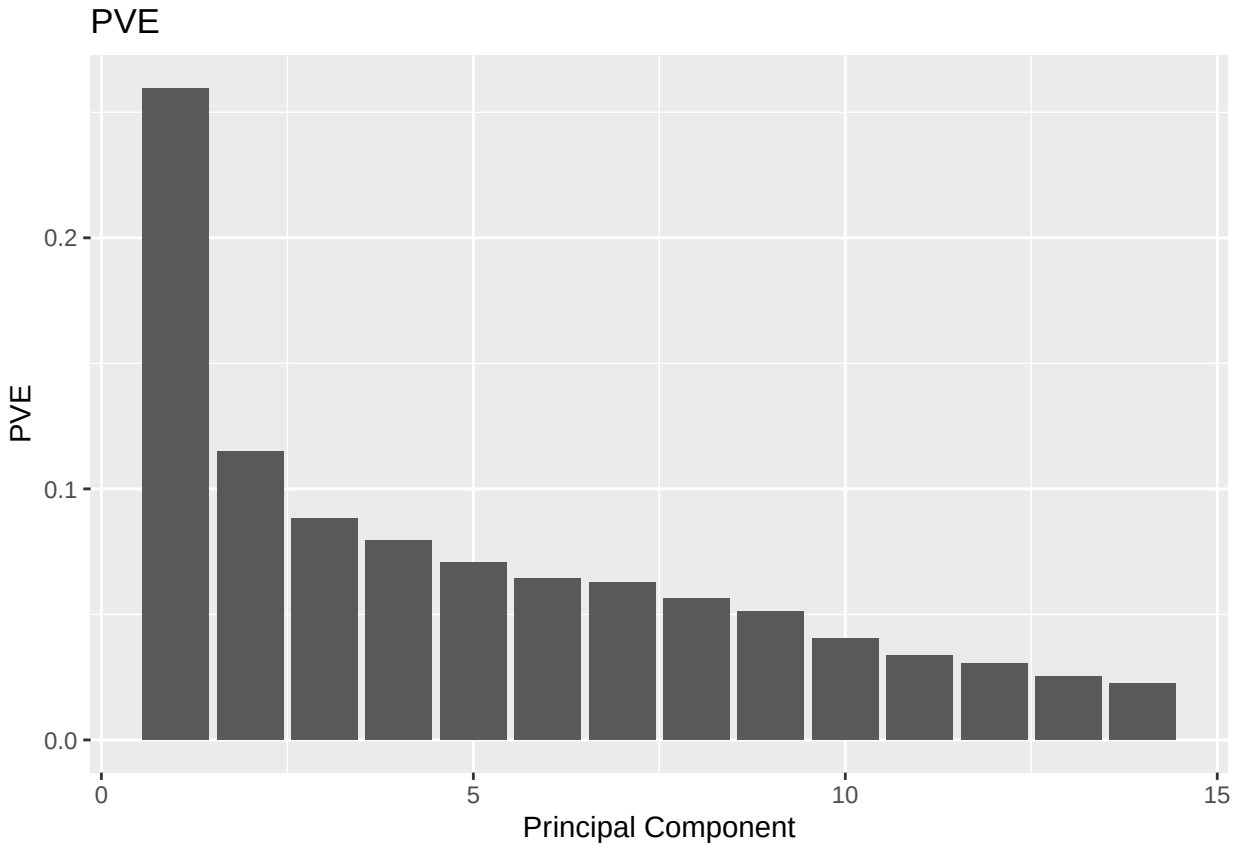
Plotten Sie die erste und zweite Hauptkomponente als Scatterplot. Färben Sie die Punkte gemäß ihrer Klasse (Disease) ein.

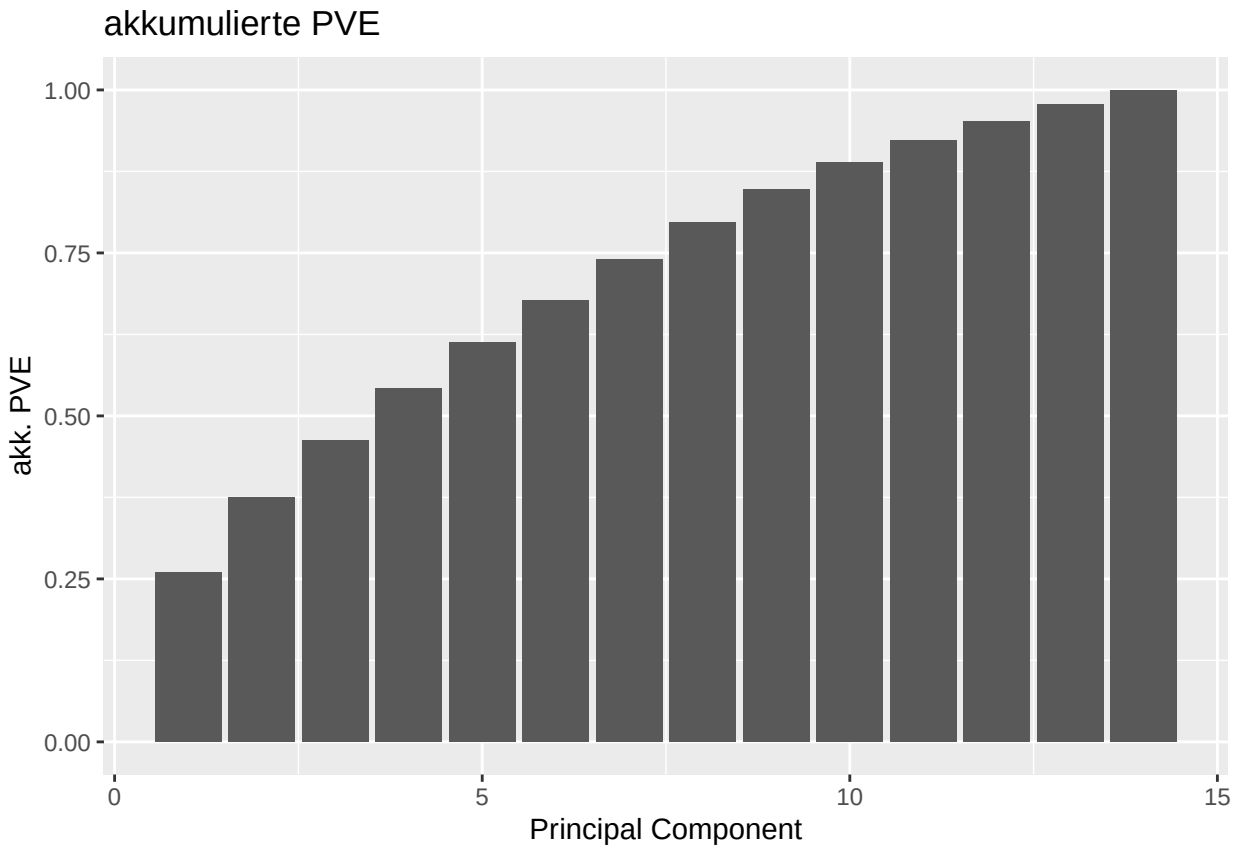
```r
autoplot(pca_res, data=df_pca, color="goal")
```




**PVE**

**Plot** Plotten Sie die Proportion of Variance explained (PVE) für jede Hauptkomponente sowie die akkumulierte PVE.


PVE
PVE
0.0
0.1
0.2
0
5
10
15
Principal Component


akkumulierte PVE
akk. PVE
0.00
0.25
0.50
0.75
1.00
0
5
10
15
Principal Component

**Wichtige Hauptkomponenten** Wie viele Hauptkomponenten erklären mehr als 50% der Varianz?

Möglicherweise tragen bei Ihrem Ergebnis die letzten Hauptkomponenten keine Varianz mehr bei. Überlegen Sie, woran das liegen könnte.

```
## Importance of components:
##                             PC1    PC2     PC3    PC4     PC5     PC6     PC7
## Standard deviation     1.9067 1.2694 1.11034 1.0550 0.99422 0.94883 0.93717
## Proportion of Variance 0.2597 0.1151 0.08806 0.0795 0.07061 0.06431 0.06274
## Cumulative Proportion  0.2597 0.3748 0.46283 0.5423 0.61293 0.67724 0.73997
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.88959 0.84724 0.75262 0.68632 0.65265 0.59533 0.55976
## Proportion of Variance 0.05653 0.05127 0.04046 0.03365 0.03043 0.02532 0.02238
## Cumulative Proportion  0.79650 0.84777 0.88823 0.92188 0.95230 0.97762 1.00000
```

Hier erklären 4 Hauptkomponenten mehr als 50% der Varianz (siehe Zeile "Cumulative Proportion"). Die letzten Hauptkomponenten tragen noch zur Varianz bei, allerdings nur sehr wenig. In der PCA werden die Hauptkomponenten so angeordnet, dass die erste die höchste Varianz im Datensatz erklärt, die zweite die zweithöchste, usw. Die späteren Hauptkomponenten erklären sukzessive immer weniger Varianz im Vergleich zu den vorherigen.

Auf Datensätze bezogen gibt es oft Muster und Strukturen, die von einigen dominanten Merkmalen bzw. Hauptkomponenten gut erklärt werden können. Die letzten Hauptkomponenten repräsentieren feinere Details, die somit weniger Varianz erklären.